Docket No.: OIN 1004-1US
(PATENT)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:
Bart A. Meltzer

Application No.: 09/173,858 | Confirmation No.: 4734

Filed: October 16, 1998 | Art Unit: 2178

For: DOCUMENTS FOR COMMERCE IN TRADING PARTNER NETWORKS AND INTERFACE DEFINITIONS BASED ON THE DOCUMENTS | Examiner: C. L. T. Huynh

INVENTOR'S DECLARATION UNDER RULE 131 PROVING REDUCTION TO PRACTICE ON OR BEFORE JANUARY 21, 1998

I, Matthew Fuchs, declare as follows:

1. I am a former employee of CN Group, Veo and Commerce One and a named inventor. All of the statements made herein are my own personal knowledge or of my own opinion, based on my training and experience, except where stated on information and belief. I could competently testify thereto if called as a witness.

2. This declaration is given in support of the application entitled “Documents for Commerce in Trading Partner Networks and Interface Definitions Based on the Documents,” U.S. patent application serial number 09/173,858, filed on October 18, 1998, and any other applications in which it might be submitted. In the course of preparation of this declaration, I looked at the documents listed in the attached list of exhibits, a set of patent claims, and this patent application. I am informed and believe that some of the exhibits were obtained from archives maintained by my former colleague Kevin Hughes.

3. In beginning in 1997 and continuing through 1998, I worked for CN Group, which became Veo and merged into Commerce One. In this declaration, "CN Group" and "Veo" are used interchangeably without any effort to track down the date or circumstance of the name change. Commerce One is mentioned in passing, as the events discussed in this declaration preceded the date on which merger of Veo into Commerce One became effective. Years ago, I received stock and/or stock options in Commerce One, which I sold at a profit before Commerce One declared bankruptcy.

4. I have been interested in the disposition of Commerce One's patents and patent applications, because they are very significant to business-to-business ecommerce. I am informed and believe that the Commerce One patents and patent applications were auctioned by a bankruptcy court and are now assigned to Open Invention Network ("OIN"), an organization formed by IBM, Phillips, Sony, Red Hat and Novell (and more recently joined by NEC), which has the web site www.openinventionnetwork.com. I am not involved in OIN, but I appreciate their stated dedication to acquire patents and offer them royalty free to promote Linux and spur innovation globally.

5. I have worked with OIN's counsel Mr. Beffel more than one occasion, to supply declarations that describe work I did as an inventor. We have had lunch and Mr. Beffel has promised to buy me and my wife dinner, in appreciation for my time.

6. Development of the technology disclosed in this patent application responded to a widely felt need for an object-oriented architectural framework for Internet commerce. The perceived need and efforts by IBM, Microsoft and others were described in an article by Tenenbaum, Jay M., Tripatinder S. Chowdhry and Kevin Hughes, "Eco System: An Internet Commerce Architecture" Computer May 1997: 48-55 ("Eco System article", submitted herewith, Exhibit A). The 1997 article proposed to use the conventional CORBA technology, which proved impractical.

7. CN Group shifted from developing a CORBA architecture and to a document-based transaction interface architecture months before W3C published XML as a

recommendation in February 1998. CN Group's change in approach is explained in a second article by Glushko, Robert J., Jay M. Tenenbaum, Bart Meltzer, "An XML Framework for Agent-based E-commerce" Communications of the ACM, Vol. 42, No. 3, pp.106-109 & 111-114 (March 1999) ("XML Framework", submitted herewith, Exhibit B) that published after this application was filed. Note that several of the figures in the XML Framework article resemble figures used in the patent application.

8. Before this patent application was filed, we developed multiple versions of data structures that established how to define a document-based transaction interface architecture. We could see that these data structures, when placed in memory, would define an interface to a transaction using one or more input documents and one or more output documents. The interface defined by these data structures was practical and workable, in contrast to the CORBA architecture proposed in the 1997 article. We understood that these data structures also could drive a code generating process that would generate Java class definitions for internal data manipulation and translators between XML and Java for marshalling and unmarshalling data between external and internal formats. Unlike CORBA, the document-based transaction interface architecture involved programs communicating with one another in a human readable external format, while using a machine-friendly internal format. So-called marshalling and unmarshalling components translated between XML and Java.

9. This declaration presents two of the data structure versions developed to define a document-based transaction interface architecture, one for a demonstration and one presented during a conference. The earlier version showed a multi-stage transaction and specified several document exchanges as parts of the transaction. It is dated January 3, 1998, which is well in advance of a demonstration that was scheduled for January 21, 1998 (Exhibit F), which demonstration may have been postponed. The conference version was presented on July 25, 1998 (Exhibit I, slide 30).

10. The XML Framework article and this patent application explain how CBL was used to standardize interpretation data for XML documents. The resulting XML documents

are useful as input and output documents referenced in a data structure that defines a document-based transaction interface architecture. However, a document-based transaction interface need not be implemented using CBL and CBL can be used in other ways that do not involve using a document-based transaction interface.

11. Submitted with this declaration are files from three versions of CBL and infrastructure supporting transactions, which I am informed and believe were found in directories named 072, 075 and "ingram/01," and which bore file date stamps and internal documentary dates before January 21, 1998.

(a) An index.html file (Ex. C) from directory 072 lists many modules as included in version 0.7.2, before the end of 1997. The index.html file also provides a limited description of module testing and validation that had been completed before the end of 1997.

(b) I am informed and believe that reprinted files in Exhibit D come from the directory 072, even though some of them have earlier version numbers, such as catentry.dtd version 0.6 and cblcat.dtd version 0.6.1..

(c) The modules in directory 075 are internally documented as belonging to version 0.8, but I am informed and believe that another version of the modules appears in a directory 080, with later dates. I am informed and believe that reprinted files in Exhibit E come from directory 075.

(d) I am informed and believe that reprinted files in Exhibit H come from directory ingram/01.

(e) Before January 21, 1998, files and modules from the three versions of CBL and infrastructure supporting transactions represented in Exhibits C-E and H were available for use in a demonstration.

12. Exhibit D consists of files dated from November 2 through December 5, 1997 from the directory 072. It includes, in the file "command.dtd" dated December 5, 1997, definitions of an ENTITY and ELEMENTs used for registering documents and services in a registry. The ENTITY is named "command" and the ELEMENTs include "command.set", "register.document" and "register.service". Registering documents and services was part of the infrastructure that we developed for using a choreographed exchange of documents to conduct a transaction. That is, as of December 5, 1997, Exhibit D already includes the basic infrastructure defining the registration of documents and services in a registry.

13. Submitted as Exhibit F is a plan for preparing for a demonstration to Ingram Micro, entitled "Requirements and Tasks for the January Demo, (Updated 1/6/98 by Kenneth)". The plan called for "1-3 clients which run in a browser" and "1 server process". "All information exchange is done via CBL/XML streams …" The demonstration scenario is described as follows, from pages 4 and 6 of the plan:

> "1. Market Description. Ingram creates **imdesc.xml** and publicizes it. imdesc.xml refers to ingram.xml.
>
> "2. IBM, DEC, COMPUSA, and Fry's send market participant info to Ingram so they can be approved and listed in the market's directories.
>
> "ibm.xml, dec.xml, compu.xml, frys.xml
>
> "Ingram acks and registers the documents.
>
> "3. IBM and DEC send product descriptions to Ingram, think.xml and hinote.xml. Ingram acks and from those product descriptions produces its own catalogue entries for them. Those cat entries are TBS. (May need multiple files for the multiple configs of these products.)
>
> "4. COMPUSA and Fry's inquire about laptops that Ingram has. NEED DETAILS. TBS: the information request docs and the responses.
>
> "5. We pretend that both COMPUSA and Fry's buy some of each laptop.
>
> "6. IBM and DEC inquire about laptops that Ingram has and has sold (= laptops in the channel?). TBS: the information request docs and the responses.
>
> ***

"(More details)

"1) Setting up the "master node" in the channel --defining Ingram's role and the services it will be providing (e.g registration, integrated catalogs, part number mapping, handling queries about prices and inventories, ordering)

"2) Registering companies to participate in the channel as either (a) manufacturers or (b) resellers [this means describing themselves (core metadata), their services (e.g., their catalog schemas), their forms (their business document schemas)]

"3) The reseller's view of the channel:

a) show me the integrated catalog

b) here's a part number: what is everyone else calling it

b) show me my price list

"4) The manufacturer's view of the channel:

a) show me (my items in) the channel

b) here's a part number: what is everyone else calling it

c) what prices are being charged for this part"

The plan for demonstration, on pages 4-5, lists many components, including ".mod" modules and ".dtd" data type definitions for XML documents. Components with names matching all of the names listed in the plan appear in the attached reprints from the 072, 075 and ingram/01 directories. Note that the scenario point # 1 is, "Ingram creates imdesc.xml and publicizes it."

14. The file "imdesc.xml" (Exhibit G), which I am informed and believe comes from the "ingram/01" directory, is one version of a data structure that defines an interface to a transaction that includes multiple input documents and output documents. The file is written in XML and points to ".dtd" definitions of input and output documents for various services related to "Ordering and Fulfillment". The file includes, in part, the text:

```
<!-- imdesc.xml Version:  0.1 -->
<!-- Purpose:  marketplace description for Ingram Micro demo -->
<!-- Terry Allen  2 Jan 1998 -->
<!-- Copyright 1998 CNgroup, Inc. -->
<?xml version="1.0"?>
<!DOCTYPE market.description SYSTEM "imarkdsc.dtd">
<market.description>
```

```
<market.name>Ingram Micro Online Sales Network
</market.name>
<market.operator>
<market.operator.name>Ingram Micro Inc.
</market.operator.name>
<market.participant.info.pointer>
   <xll.locator urllink="ingram.xml">Ingram's Market Participant Info
   </xll.locator>
</market.participant.info.pointer>
</market.operator>
<service.set>
   <service>
   <service.name>Ordering and Fulfillment
   </service.name>
  <service.function.sequence>
   <service.function>
      <doctype from.party="any" to.party="ingram">order.dtd</doctype>
      <doctype from.party="ingram" to.party="any">ack.dtd</doctype>
   </service.function>
   <service.function>
      <doctype from.party="ingram" to.party="any">invoiceo.dtd</doctype>
      <doctype from.party="any" to.party="ingram">ack.dtd</doctype>
   </service.function>
   <service.function>
      <doctype from.party="ingram" to.party="any">shipnote.dtd</doctype>
   </service.function>
   <service.function>
      <doctype from.party="any" to.party="ingram">paynoteo.dtd</doctype>
      <doctype from.party="ingram" to.party="any">ack.dtd</doctype>
   </service.function>
 </service.function.sequence>
</service>
<service> …
```

The bold facing is added for emphasis. The internal documentation date of this file is January 2, 1998. I am informed and believe that the file date stamp indicates that the file was last changed on January 4, 1998 at 3:47 a.m. Both dates are well in advance of the scheduled demonstration on January 21, 1998.

15. The imdesc.xml file, in the excerpt above, describes service functions for "Ordering and Fulfillment" that are registered to be accessible through Ingram's URL. One

service receives an order and acknowledges it. Another generates an invoice and expects an acknowledgment. A third generates a shipping notice, without expecting an acknowledgement. Another receives a payment notice and acknowledges it. Each of these services is accessible through an interface that accepts an XML input document that conforms to a ".dtd" file and, typically, returns an XML acknowledgement document.

16. Before January 21, 1998, we had sufficiently worked with imdesc.xml to recognize and understand that it would serve its intended purpose of defining a document-based transaction interface that can include a series of related document exchanges. This file was one of many that were under development. The "index.html" (Ex. C) confirms that these files were tested during development. Exhibits D, E and H demonstrate the availability of the supporting files called for by the demonstration plan. By comparison of Exhibits D, E and H to the demonstration plan (Ex. F), one sees that all of the files listed as desired for the demonstration were available before January 21, 1998. Our intended use of the "imdesc.xml" file was as an interface definition. (Ex. F, quoted above) In January 1998, we correctly understood that the imdesc.xml data structure would work for its intended purpose. Later versions of our interface definition data structure, such as the version that appears in our patent application, generally followed the document interface definition pattern of "imdesc.xml". The interface definition version (Exhibit I) that was publicly disclosed at a conference on July 25, 1998, at the International Workshop on Component-based Electronic Commerce was held at the Fisher Center for Management & Information Technology, Haas School of Business, UC Berkeley also followed the same pattern.

17. The July 25 version, Slide 30, Exhibit I included:

```
<service>
<service.name>Order Service</service.name>
    <service.location>www.veosystems.com/order</service.location>
    <service.op>
        <service.op.name>Submit Order</service.op.name>
        <service.op.inputdoc>po.dtd</service.op.inputdoc>
        <service.op.outputdoc>poack.dtd</service.op.outputdoc>
    </service.op>
```

```
    < service.op>
        < service.op.name>Track Order</service.op.name>
        <service.op.inputdoc>request.track.dtd<service.op.inputdoc>
        <service.op.outputdoc>response.track.dtd<service.op.outputdoc>
    </service.op>
</service>
```

A very similar version appears in the patent application on page 45, which I am informed and believe also appears as CD-ROM appendix LISTING 5, after reformatting of specification.

18. Our January 1998 understanding that this data structure would work for its intended purpose was subsequently validated by others, who adopted our approach to interface definition. For instance, the later W3 "note" dated March 15, 2001 describing WSDL version 1.1 followed the same document-based interface pattern. In Exhibit J, taken from http://www.w3.org/TR/wsdl, section 1.2, WSDL Document Example, we see a simplified version of our interface definition, which involves a single document exchange, as opposed to a series of related document exchanges:

```
<portType name="StockQuotePortType">
   <operation name="GetLastTradePrice">
      <input message="tns:GetLastTradePriceInput"/>
      <output message="tns:GetLastTradePriceOutput"/>
   </operation>
</portType>
```

19. Next, we compare the two versions of our data structure to the pending claims. Singularly and collectively, the versions of data structures, which were stored in memory accessible to at least one node on a network, included the characteristics and features described below.

Claim 1: In Exhibits G-I, both imdesc.xml and Slide 30 depict machine readable interface specifications. The imdesc.xml file typically is found in a file directory on a

machine readable storage media. The PowerPoint Slide 30 appears to have been taken from a computer file similar to imdesc.xml and pasted into the presentation. The archive for imdesc.xml and the PowerPoint presentation both were stored on machine readable storage media. Both data structures define an interface to a transaction process. The imdesc.xml defines an interface with several service "functions". Slide 30 defines multiple service "operations". In imdesc.xml, one of the input documents is an "order"; in Slide 30, there is a "po", which is short for purchase order. In both transaction interface definitions, an acknowledgement is sent, an "ack" in the earlier version and a "poack" in the later version. The input document definitions are referenced by "order.dtd", "invoiceo.dtd", "paynoteo.dtd", "po.dtd" and "request.track.dtd", which are data type definition (dtd) files. The output document definitions are referenced by "ack.dtd", "poack.dtd" and "response.track.dtd". The January 1998 demonstration scenario (Ex. F, quoted above) makes it clear that these interface definitions were published to nodes on a network that might desire to invoke the functions for which interfaces were defined. The context of the slides in Exhibit I makes it clear that this interface was hosted and accessible to a plurality of nodes on a network in the development environment from which it was borrowed for the PowerPoint presentation.

Claim 2: The data type specifications in the exemplary "order.dtd" (Ex. E) include at least one logical structure. Similarly, slides 25-29 (Ex. I) depict using logical structure building blocks to construct documents such as a purchase order (po.dtd). In general, a data type definition file (dtd) will include data type specifications for at least on logical data structure.

Claim 3: The data type specifications for the country field of the exemplary "order.dtd" (Ex. E) include at least one data structure mapping predefined sets of storage units for a particular logical structure in the definitions to respective entries in a list. The country code field is supported by a list in "codes.mod" (the codes module), which is incorporated into addresso.mod which, in turn, is part of order.dtd. With a bit of tracing from imdesc.xml, one can see where the reusable lists were part of the system. Turning to

the July 25, 1998 presentation, slides 27-28 similarly illustrate use of country codes in a purchase order, defined by the file "po.dtd".

Claim 4: In Exhibit D, the ENTITY "command" and the related ELEMENTs support registering definitions of documents and services in a repository in memory accessible to at least one node. The definitions of documents and services include logical structures and interpretation information for logical structures. In Exhibit I, Slide 23 depicts a service registration document in a registry accessible to discovery nodes on a network. Slide 31 indicates that CBL was already in use for demonstration applications, Project Seitai and GSA catalog interoperability. We found it very useful to post our interface definitions for programmers to rely on during development, such as the imdesc.xml interface definition.

Claim 5: Looking through files from the ingram/01 directory, which are reproduced in Exhibit H, one finds the data type definition corresponding to the sample interface definition imdesc.xml. Beginning with imdesc.xml on page 32 of 66, one sees reference to "imarkdsc.dtd" in the DOCTYPE statement of line six. This data type definition appears on page 29 of 66, and incorporates by reference the "isrvprim.mod" module that appears on page 54 of 66. The isrvprim.mod defines elements of imdesc.xml, including service.name, service.location.pointer, and service.function. Thus, the sample "imdesc.xml" machine readable specification of Exhibit G complies with the imarkdsc.dtd definition of an interface document including logical structures for storing identifiers (e.g., service.name and/or input doctype) and references to definitions (dtd's) of input and output documents. Similarly, in Exhibit I, slide 30, identifiers of particular transactions appear as "Submit Order" and "Track Order." The references to definitions of the input and output documents are dtd file names. The "Loose Coupling" via Shared Document Definitions slide 22 makes it clear that the definition in slide 30 is a document that complies with a dtd, as in the ingram/01 files.

Claim 6: Both the "imdesc.xml" Exhibit G and the Slide 30 in Exhibit I specify one or more transactions supported by the interface.

Claim 7: Both the "imdesc.xml" Exhibit G and the Slide 30 in Exhibit I include references to documents used in the particular transactions, specifying the document dtd's by name instead of providing a copy of them.

Claims 8-12: The input and output documents in both versions are depicted as XML documents. Generally, XML documents compliant with the February 1998 recommendation can be parsed or unparsed data. XML documents may encode text characters and the text characters may provide a natural language word. XML documents generally include markup data to identify sets of storage units.

Claim 13: The files reprinted in Exhibits D, E and H from directories 072, 075 and ingram/01 store document types that can be used in a plurality of transactions. Commands identified above were used to register these document types in a repository. Slides 25-29 depict a library of document types stored in a repository. The dtd definitions for the documents used in the transactions are referenced by names. For instance, in "imdesc.xml" we see the "ack.dtd" as the output document for several service functions.

Claim 14: The collections of files from directories include "imarkprt.dtd" and "imarkdsc.dtd" for market participant information and market description information. These modules are called out in Slide 28 of Exhibit I by slightly different and more easily readable names, "markpart.dtd" and "markdesc.dtd". Some of these document types include identifications of participant processes.

Claim 15: The dtd files referenced in "imdesc.xml" and Slide 30, e.g., order.dtd and ack.dtd, are recognizable as compliant with a standard Extensible Markup Language XML.

Claim 16: The file "imdesc.xml" and the code excerpt in Slide 30 are recognizable as compliant with a standard Extensible Markup Language XML.

Claim 61: In the course of creating "imdesc.xml" and Slide 30, the authors of those documents went through the process of defining a machine readable interface

definition including an input and output document. The versions of the transaction interface definition data structure were provided to network nodes that requested the definition data structures.

Claim 62: The data type specifications in the exemplary "order.dtd" include at least one logical structure. Similarly, slides 25-29 depict using logical structure building blocks to construct documents such as a purchase order (po.dtd). In general, a data type definition file (dtd) will include data type specifications for at least on logical data structure.

Claim 63: The data type specifications for the country field of the exemplary "order.dtd" (Ex. E) include at least one data structure mapping predefined sets of storage units for a particular logical structure in the definitions to respective entries in a list. The country code field is supported by a list in "codes.mod" (the codes module), which is incorporated into addresso.mod which, in turn, is part of order.dtd. With a bit of tracing from imdesc.xml, one can see where the reusable lists were part of the system. Use of country codes in addresses is a feature of CBL that extended the XML recommendation – the XML recommendation uses codes for language identification and not for addresses. *See*, W3C Recommendation, Extensible Markup Language (XML) 1.0, § 2.12 (Feb. 10, 1998) accessed at http://www.w3.org/TR/1998/REC-xml-19980210#dt-app. Turning to the July 25, 1998 presentation, slides 27-28 similarly illustrate use of country codes in a purchase order, defined by the file "po.dtd".

Claim 64: Slide 23 (Ex. I) depicts a service registration document in a registry accessible to nodes on a network. Slide 31 indicates that CBL was already in use for demonstration applications, Project Seitai and GSA catalog interoperability. We found it very useful to post our interface definitions for programmers to rely on during development, such as the imdesc.xml interface definition. In the January 1998 and earlier materials (Exs. D, E and H), we have printed some libraries of logical structures that we were using then. The Slides indicate that the libraries of logical structures were still in use six months later, when the conference presentation was made in July 1998.

Claim 65: Looking through files from the ingram/01 directory (Ex. H) one finds how imdesc.xml has been defined. Beginning with imdesc.xml on page 32 of 66, one sees reference to "imarkdsc.dtd" in the DOCTYPE statement of line six. This data type definition appears on page 29 of 66, and incorporates by reference the "isrvprim.mod" module that appears on page 54 of 66. The isrvprim.mod defines elements of imdesc.xml, including service.name, service.location.pointer, and service.function. Thus, the sample "imdesc.xml" machine readable specification of Exhibit G complies with the imarkdsc.dtd definition of an interface document including logical structures for storing identifiers (e.g., service.name and/or input doctype) and references to definitions (dtd's) of input and output documents. Similarly, in Exhibit I, slide 30, identifiers of particular transactions appear as "Submit Order" and "Track Order." The references to definitions of the input and output documents are dtd file names. The "Loose Coupling" via Shared Document Definitions slide 22 makes it clear that the definition in slide 30 is a document that complies with a dtd, as in the ingram/01 files.

Claims 66-70: The input and output documents in both versions (code and presentation) are depicted as XML documents. Generally, XML documents compliant with the February 1998 recommendation can be parsed or unparsed data. XML documents may encode text characters and the text characters may provide a natural language word. XML documents generally include markup data to identify sets of storage units.

Claim 71: The dtd files referenced in "imdesc.xml" and Slide 30 are recognizable as compliant with a standard Extensible Markup Language XML.

Claim 72: Our development work in 1997-98 included applying parsers to input documents, generating one or more Java event signals in response to the logical structure of the input documents, and having event listener programs respond to the event signals.

20. I am informed and believe that Examiner Huynh has questioned whether versions of CBL prior to the publicly released version 1.1 worked. In fact, CBL worked for

its intended purpose for many months and many versions prior to the September 1998 release of public version 1.1.

I declare under penalty of perjury of the laws of the United States of America that the foregoing is true and correct. I make this declaration with the understanding and knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 U.S.C. and that making willful false statements would jeopardize the validity of my application and any patents issuing thereon.

Executed this 21th day of July, 2008 in Palo Alto, CA.

Matthew Fuchs.

Exhibit List

A. Tenenbaum, Jay M., Tripatinder S. Chowdhry and Kevin Hughes, "Eco System: An Internet Commerce Architecture" Computer May 1997: 48-55

B. Glushko, Robert J., Jay M. Tenenbaum, Bart Meltzer, "An XML Framework for Agent-based E-commerce" Communications of the ACM, Vol. 42, No. 3, pp. 106-109 & 111-114 (March 1999)

C. "index.html" from cbl/072 directory (file date stamped in 1997)

D. Selected files from cbl/072 directory (date stamped in 1997)

E. Selected files from cbl/075 directory (date stamped before January 21, 1998)

F. "Requirements and Tasks for the January Demo, (Updated 1/6/98 by Kenneth)", file named "demo_req_tasks.html" from Veo/web/dev/documents/old/demo directory (date stamped before January 21, 1997)

G. "imdesc.xml" from cbl/ingram/01 directory (date stamped before January 21, 1997)

H. Selected files from cbl/ingram/01 directory (date stamped before January 21, 1997)

I. Glushko, Robert J., *Implementing Domain-specific Commerce Languages with a Common Business Library* (delivered July 25, 1998) accessed at http://groups.haas.berkeley.edu/citm/conferences/cec/Presentations/Session3/glushko.pdf on October 26, 2006

J. Excerpts from W3C "note" WSDL version 1.1 (March 15, 2001) accessed at http://www.w3.org/TR/wsdl